

Principal ideal languages and synchronizing automata

Vladimir V. Gusev, Marina I. Maslennikova, Elena V. Pribavkina

Ural Federal University, Ekaterinburg, Russia
 vl.gusev@gmail.com, maslennikova.marina@gmail.com,
 elena.pribavkina@usu.ru

Abstract. We study ideal languages generated by a single word. We provide an algorithm to construct a strongly connected synchronizing automaton for which such a language serves as the language of synchronizing words. Also we present a compact formula to calculate the syntactic complexity of this language.

Keywords: ideal language, synchronizing automaton, synchronizing word, strongly connected automaton, syntactic complexity.

1 Introduction

Let $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ be a *deterministic finite automaton* (DFA for short), where Q is the *state set*, Σ stands for the *input alphabet*, and $\delta : Q \times \Sigma \rightarrow Q$ is the *transition function* defining an action of the letters in Σ on Q . The action extends in a natural way to an action $Q \times \Sigma^* \rightarrow Q$ of the free monoid Σ^* over Σ ; the latter action is also denoted by δ . When δ is clear from the context, we will write $q.w$ instead of $\delta(q, w)$ for $q \in Q$ and $w \in \Sigma^*$. In the theory of formal languages the definition of a DFA usually includes the set $F \subseteq Q$ of *terminal states* and an *initial state* $q_0 \in Q$. We will use this definition when dealing with automata as devices for recognizing languages. The language $L \subseteq \Sigma^*$ is *recognized* (or *accepted*) by an automaton $\mathcal{A} = \langle Q, \Sigma, \delta, F, q_0 \rangle$ if $L = \{w \in \Sigma^* \mid \delta(q_0, w) \in F\}$. We also use standard concepts of the theory of formal languages such as regular language, minimal automaton, etc. [6]

A DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ which leaves the automaton in unique state no matter which state in Q it is started to read: $\delta(q, w) = \delta(q', w)$ for all $q, q' \in Q$. Such word is said to be *synchronizing* (or *reset*) for the DFA \mathcal{A} . This notion has been widely studied since the work of Jan Černý [2] in 1964. He conjectured that any synchronizing DFA with n states possesses a synchronizing word of length at most $(n-1)^2$. This conjecture is widely open and is considered one of the most longstanding open problems in the combinatorial theory of finite automata. Various techniques were developed to approach this conjecture. For more information on synchronizing automata we refer the reader to the thorough survey by Mikhail Volkov [9]. In this paper we focus on language theoretic aspects of the Černý conjecture and related questions.

Recall that a language L over Σ is called *ideal* if $L = \Sigma^* L \Sigma^*$. By $\text{Syn}(\mathcal{A})$ we denote the language of all words synchronizing \mathcal{A} . It is easy to see that $\text{Syn}(\mathcal{A})$ is an ideal language. In what follows we consider only ideal regular languages. It was observed in [4] that the minimal deterministic automaton \mathcal{A}_L recognizing an ideal regular language L is synchronizing, and $\text{Syn}(\mathcal{A}_L) = L$. Interesting question arises: how many states the smallest automaton \mathcal{A} such that $\text{Syn}(\mathcal{A}) = L$ may have? This question was posed in [4] and the notion of reset complexity was introduced. The *reset complexity* $rc(L)$ of an ideal language L is the minimal possible number of states in a synchronizing automaton \mathcal{A} such that $\text{Syn}(\mathcal{A}) = L$. For brevity we will call the corresponding automaton MSA (*minimal synchronizing automaton*). The Černý conjecture can be stated in terms of reset complexity as follows. Let ℓ be the minimal length of words in an ideal language L , then $rc(L) \geq \sqrt{\ell} + 1$. Even a lower bound $rc(L) \geq \frac{\sqrt{\ell}}{C}$ for some constant C would be a major breakthrough.

From descriptive complexity point of view it is interesting to compare reset complexity with the classical state complexity. The *state complexity* $sc(L)$ of a regular language L is the number of states in \mathcal{A}_L . In [4] it was observed that $rc(L) \leq sc(L)$. Also in [4] it was shown that in some cases $rc(L)$ can be exponentially smaller than $sc(L)$. In particular, it means that the description of an ideal language L by means of an automaton \mathcal{A} for which $\text{Syn}(\mathcal{A}) = L$ can be exponentially more succinct than the “standard” description via minimal automaton recognizing L . The minimal automaton of an ideal regular language always has a sink state (a state fixed by all letters), whereas the corresponding MSA may be strongly connected, which means that for any two states p and q ($p \neq q$) there exists a word mapping p to q . Automata with the sink state and strongly connected automata are essential for the Černý conjecture, since it was shown in [10] that it is enough to prove this conjecture for each of the two classes of automata. Thus, we may ask whether it is always possible to construct a strongly connected synchronizing DFA for which L serves as the language of synchronizing words.

We begin to approach this question by considering principal ideal languages, i.e. ideal languages generated by a single word. A principal ideal language is a partial case of a finitely generated ideal language. The latter languages viewed as languages of synchronizing words were considered in [7] and [8].

In section 2 we answer the uniqueness question that was posed in [4]. The question is whether the uniqueness of an MSA takes place within the class of strongly connected automata. The answer is negative. For the language $L = \Sigma^* a^{n-1} b \Sigma^*$ there exist two different strongly connected automata with $n + 1$ states over $\Sigma = \{a, b\}$ yielding the minimum of reset complexity for L .

In section 3 we provide an algorithm to construct a strongly connected synchronizing automaton whose language of synchronizing words is generated by a single word.

In section 4 we consider some algebraic properties of principal ideal languages. In particular, we establish the connection between the syntactic semigroup of such a language and the transition semigroup of a synchronizing automaton

for which this language serves as the language of reset words. Also, we find a compact formula for calculating the syntactic complexity of a principal ideal language.

2 On uniqueness question of an MSA

It is well-known that the minimal automaton \mathcal{A}_L recognizing a given language L is unique up to isomorphism. The same fact does not hold for an MSA, see [4]. But the question, whether the uniqueness takes place within the class of strongly connected automata, remained open. Here we answer this question in the negative.

Consider the language $L = \Sigma^* a^{n-1} b \Sigma^*$ over $\Sigma = \{a, b\}$. In [4] it was shown that $rc(L) = n + 1$. We present two different strongly connected automata with $n+1$ states for which L serves as the language of synchronizing words. For clarity the corresponding automata \mathcal{A}_6 and \mathcal{B}_6 with six states are shown on Fig. 1.

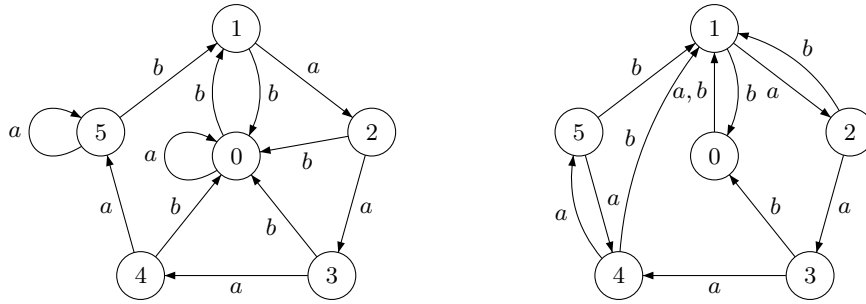


Fig. 1. Automata \mathcal{A}_6 and \mathcal{B}_6 .

The transition function δ of the first DFA \mathcal{A}_{n+1} is defined as follows:

$$\delta(i, a) = \begin{cases} i+1 & \text{if } 0 < i < n, \\ 0 & \text{if } i = 0, \\ n & \text{if } i = n, \end{cases} \quad \delta(i, b) = \begin{cases} 0 & \text{if } 0 < i < n, \\ 1 & \text{if } i = 0 \text{ or } i = n. \end{cases}$$

Note that \mathcal{A}_{n+1} is strongly connected. Indeed, the states $1, 2, \dots, n$ appear in a cycle marked by the word $a^{n-1}b$, furthermore $0 \cdot b = 1$ and $1 \cdot b = 0$. Now we need to check, that the language of words synchronizing the automaton \mathcal{A}_{n+1} coincides with $\Sigma^* a^{n-1} b \Sigma^*$. The standard tool for finding the language of synchronizing words of a given DFA $\mathcal{A} = \langle Q, \delta, \Sigma \rangle$ is the *power automaton* $\mathcal{P}(\mathcal{A})$. Its state set is the set \mathcal{Q} of all nonempty subsets of Q , and the transition function is defined as a natural extension of δ on the set $\mathcal{Q} \times \Sigma$ (the resulting function is also denoted by δ), namely $\delta(S, a) = \{\delta(s, a) \mid s \in S\}$ for $S \subseteq Q$ and $a \in \Sigma$. The automaton $\mathcal{P}(\mathcal{A})$ recognizes $\text{Syn}(\mathcal{A})$ provided one takes Q as the initial state and singletons as final states. It is easy to see, that if all the singletons are identified

to obtain unique sink state s , the resulting automaton still recognizes $\text{Syn}(\mathcal{A})$. Throughout the paper the term *power automaton* will refer to this modified version. The Fig. 2 shows the power automaton for the language \mathcal{A}_{n+1} (for clarity only reachable from Q subsets are shown). From the structure of $\mathcal{P}(\mathcal{A}_{n+1})$ it is easy to see that the language of synchronizing words of the automaton \mathcal{A}_{n+1} coincides with L .

Next we consider the DFA \mathcal{B}_{n+1} with $n + 1 = 2k$ and transition function δ defined by the rule

$$\delta(i, a) = \begin{cases} i + 1 & \text{if } 0 \leq i < n, \\ n - 1 & \text{if } i = n, \end{cases} \quad \delta(i, b) = \begin{cases} 0 & \text{if } i \text{ is odd and } i \neq n, \\ 1 & \text{if } i \text{ is even or } i = n. \end{cases}$$

We verify that \mathcal{B}_{n+1} is strongly connected. Indeed, the states $1, 2, \dots, n$ appear in a cycle marked by the word $a^{n-1}b$, furthermore $0.a = 1$ and $1.b = 0$. It is easily seen that, for any odd n , \mathcal{B}_{n+1} and \mathcal{A}_{n+1} are not isomorphic. For the power automaton $\mathcal{P}(\mathcal{B}_{2k})$ see the left side of Fig. 3 (again, only reachable from Q subsets are shown). It remains to construct a series of strongly connected

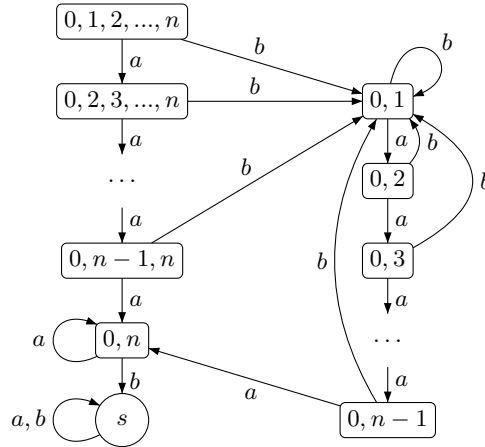


Fig. 2. The power automaton $\mathcal{P}(\mathcal{A}_{n+1})$

automata \mathcal{B}_{n+1} with $n + 1 = 2k + 1$. The transition function δ is defined as follows:

$$\delta(i, a) = \begin{cases} i + 1 & \text{if } 1 < i < n \text{ or } i = 0, \\ 0 & \text{if } i = 1 \text{ or } i = n, \end{cases} \quad \delta(i, b) = \begin{cases} 0 & \text{if } 1 < i \leq n, \\ 2 & \text{if } i = 0 \text{ or } i = 1. \end{cases}$$

The states $2, 3, \dots, n, 0$ form a cycle marked by the word $a^{n-1}b$, furthermore $0.a = 1$ and $1.a = 0$. It is easily seen that, for any even n , \mathcal{B}_{n+1} and \mathcal{A}_{n+1} are not isomorphic. The power automaton $\mathcal{P}(\mathcal{B}_{2k+1})$ consisting only of reachable from Q subsets is on the right side of Fig. 3. From the structure of the power automaton for \mathcal{B}_{n+1} it easily follows that its set of synchronizing words coincides with L .

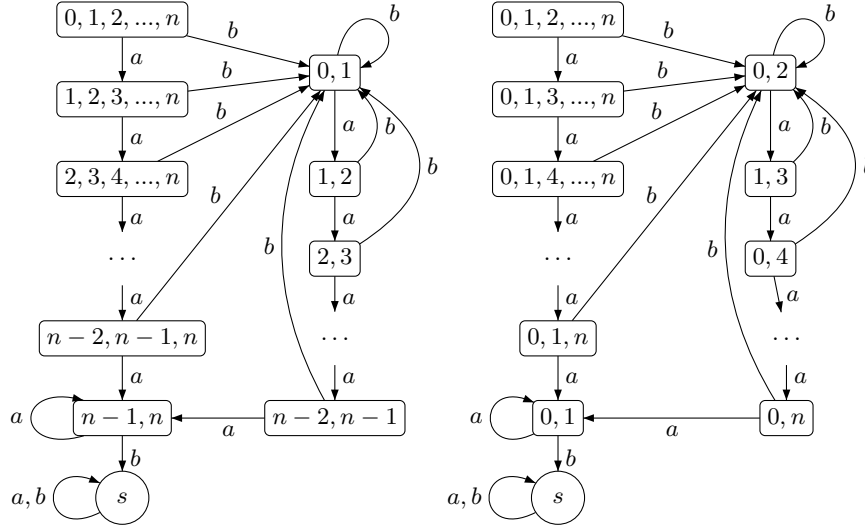


Fig. 3. The power automata $\mathcal{P}(\mathcal{B}_{2k})$ and $\mathcal{P}(\mathcal{B}_{2k+1})$

3 Algorithm

3.1 Formal description

In this section we provide an algorithm to construct a strongly connected synchronizing automaton whose language of synchronizing words is generated by a single word w . The minimal automaton recognizing this language is denoted by \mathcal{A}_w . The main idea of the construction is the following. We try to construct a strongly connected automaton such that in its *pair automaton* there is a subautomaton isomorphic to \mathcal{A}_w . Our algorithm can be applied in case of an arbitrary alphabet, but for clarity we explain it only in binary case.

Recall, that the pair automaton of a given DFA $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ is the subautomaton $\mathcal{P}^{[2]}(\mathcal{A})$ of the power automaton $\mathcal{P}(\mathcal{A})$ consisting only of 2-element subsets of Q and the sink state s .

Fix a word w over $\Sigma = \{a, b\}$. Let $|w| = n$. Without loss of generality suppose that the first letter of w is a . Denote the i -th letter of w by $w[i]$ and the prefix $w[1]w[2]\dots w[i]$ by $w[1..i]$. For any letter $x \in \{a, b\}$ by \bar{x} we denote its *complementary* letter, i.e. $\bar{a} = b$, and $\bar{b} = a$. Let us remind the construction of

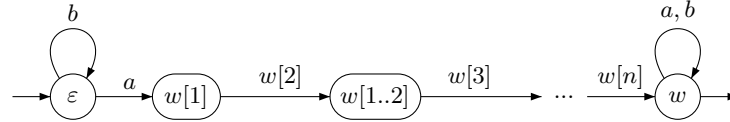


Fig. 4. The minimal DFA \mathcal{A}_w .

the minimal automaton recognizing the language $\Sigma^*w\Sigma^*$. It is well-known that this automaton has $n + 1$ states. We enumerate the states of this automaton by the prefixes of the word w so that the state $w[1..i]$ maps to the state $w[1..i + 1]$ under the action of the letter $w[i + 1]$ for all i , $0 \leq i < n$. The other letter $\overline{w[i + 1]}$ maps the state $w[1..i]$ to the state p such that p is the maximal prefix of w that appears in the word $w[1..i + 1]$ as a suffix. The state w is the sink state. The initial state is ε and the unique final state is w , see Fig.4 (the transitions labeled by complementary letters $w[i]$ are not shown).

The algorithm constructing a required strongly connected synchronizing automaton \mathcal{B} with the state set $Q = \{0, 1, \dots, n\}$ proceeds inductively. On the first step we put $Q = \{0, 1, 2\}$ and define the action of letters on the states 0 and 1. On the i^{th} step ($1 < i < n$) we add new state $i + 1$ to Q and define the transition function on the state i . On the last, n^{th} step we define the transition function on the state n . Transitions on each step are defined in such a way, that after the i^{th} step of the algorithm ($1 \leq i \leq n$) the current pair automaton has a subautomaton isomorphic to the part of the minimal automaton \mathcal{A}_w consisting of the states $\varepsilon, w[1], \dots, w[1..i]$.

Consider the first step of the algorithm. We need to associate the states ε and $w[1]$ of \mathcal{A}_w with 2-element subsets $\{p_i, q_i\}$ of $Q = \{0, 1, 2\}$. Without loss of generality we associate the state ε with the subset $\{0, 1\}$, and the state $w[1]$ with the subset $\{1, 2\}$. In the automaton \mathcal{A}_w the state ε is fixed by b and maps to $w[1]$ under the action of a . Define in \mathcal{B} the transition function on 0 and 1 in such a way that the subset $\{0, 1\}$ in \mathcal{B} is fixed under the action of b , and maps to the subset $\{1, 2\}$ under the action of a . We have four different ways to do so (see Fig.5). It is easy to see that in fact the second case is impossible

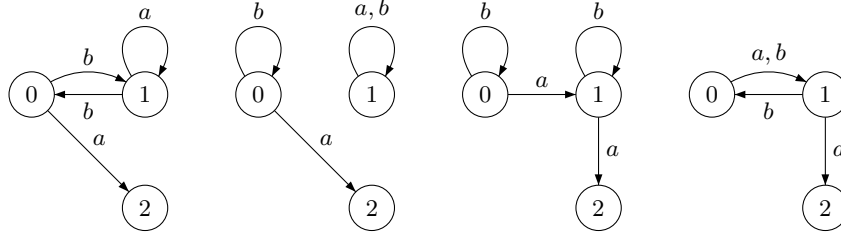


Fig. 5. Possible transitions from 0 and 1 in the automaton \mathcal{B} .

since the DFA \mathcal{B} will not be strongly connected, not even during the rest of the construction. For certainty consider the first variant, so $0.a = 2$, $0.b = 1$, $1.a = 1$, and $1.b = 0$.

Let us describe the i^{th} step of the algorithm. We have $Q = \{0, 1, \dots, i\}$. For convenience we represent the subsets $\{p_i, q_i\} \neq \{0, 1\}$ of Q as ordered pairs (p_i, q_i) with $p_i > q_i$, and the subset $\{0, 1\}$ as pair $(0, 1)$. On previous steps the states $\varepsilon, w[1], \dots, w[1..i-1]$ of \mathcal{A}_w were associated with the pairs $(p_0, q_0), (p_1, q_1), \dots, (p_{i-1}, q_{i-1})$ in $\mathcal{P}^{[2]}(\mathcal{B})$ in such a way that $p_0 = 0, p_1 = 2, p_2 = 3, \dots, p_{i-1} = i$ and the transition function on the states $0, 1, \dots, i-1$ was defined (see the dash-

dotted part on Fig. 6). We add the state $i + 1$ to Q . Next we associate the state $w[1..i]$ of \mathcal{A}_w with the pair $(i+1, q_i)$, where $q_i = q_{i-1} \cdot w[i]$, and put $i \cdot \overline{w[i]} = i+1$. It remains to define the transition $i \cdot \overline{w[i]}$. Let $w[1..j] = w[1..i-1] \cdot w[i]$, and let (p_j, q_j) be the associated pair of states in $\mathcal{P}^{[2]}(\mathcal{B})$. In the correctness section we will show that one of the equalities holds: either $q_{i-1} \cdot \overline{w[i]} = q_j$ or $q_{i-1} \cdot \overline{w[i]} = p_j$. If $q_{i-1} \cdot \overline{w[i]} = q_j$, then we put $i \cdot \overline{w[i]} = p_j$, otherwise, $i \cdot \overline{w[i]} = q_j$. The i^{th} step is illustrated on Fig. 6. The left part of the picture corresponds to the current pair automaton (only states currently associated to the states of \mathcal{A}_w are shown), the right part is the corresponding subautomaton of \mathcal{A}_w . The correspondence between states of \mathcal{A}_w and those of pair automaton is shown in dashed lines. The transitions defined on this step are shown in thick lines.

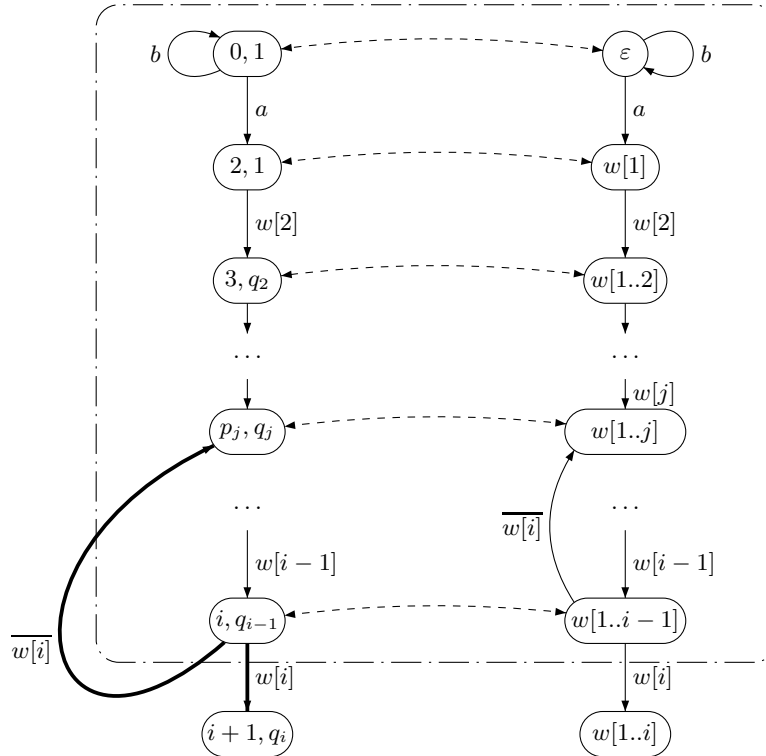


Fig. 6. Step i : Current pair automaton, and the corresponding part of \mathcal{A}_w .

On the last n^{th} step we map the state n of \mathcal{B} to the state $q_{n-1} \cdot w[n]$ under the action of the letter $w[n]$ in order to associate the state w of \mathcal{A}_w with the sink state s of $\mathcal{P}^{[2]}(\mathcal{B})$. The action of the letter $\overline{w[n]}$ is defined as before.

Obviously the language consisting of words synchronizing the subset $\{0, 1\}$ coincides with $\Sigma^* w \Sigma^*$. Since the set of words synchronizing the whole automaton

\mathcal{B} is contained in the set of words synchronizing any of its subsets, we have $\text{Syn}(\mathcal{B}) \subseteq \Sigma^* w \Sigma^*$. Let us show that the word w synchronizes \mathcal{B} . Let $0 \cdot w = 1 \cdot w = m \in Q$. We proceed inductively. Assume that for any $0 \leq k < i$ we have $k \cdot w = m$. Let us show that $i \cdot w = m$. The state i belongs to the pair (i, q) in the subautomaton of $\mathcal{P}^{[2]}(\mathcal{B})$ isomorphic to \mathcal{A}_w . Thus, the pair (i, q) is synchronized by the word w . Since $q < i$, by induction hypothesis we have $q \cdot w = m$, therefore $i \cdot w = m$. Finally, we have $n \cdot w = m$, so $w \in \text{Syn}(\mathcal{B})$. Hence $\Sigma^* w \Sigma^* \subseteq \text{Syn}(\mathcal{B})$.

Example 1. We apply our algorithm to the word $aabab$. First, build the minimal automaton recognizing $\Sigma^* aabab \Sigma^*$ (see Fig. 7). Next we show in details the

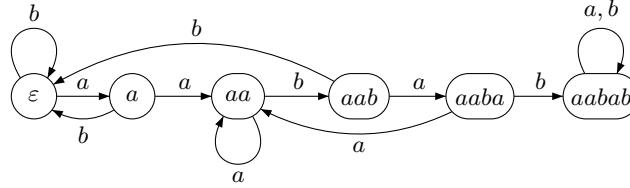


Fig. 7. The minimal DFA recognizing $\Sigma^* aabab \Sigma^*$.

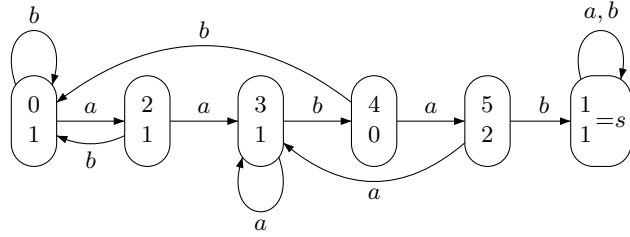


Fig. 8. The corresponding subautomaton in the pair automaton $\mathcal{P}^{[2]}(\mathcal{B})$

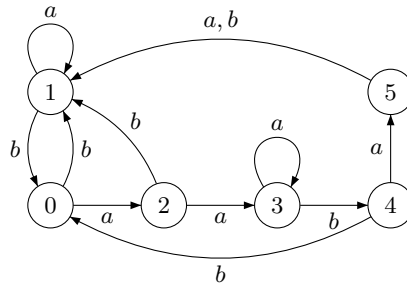


Fig. 9. Strongly connected synchronizing automaton \mathcal{B} with $\text{Syn}(\mathcal{B}) = \Sigma^* aabab \Sigma^*$.

construction of the DFA \mathcal{B} .

Step 1. We have $Q = \{0, 1, 2\}$, and $0 \cdot a = 2$, $0 \cdot b = 1$, $1 \cdot a = 1$, and $1 \cdot b = 0$.

Step 2. Add state 3 to Q and put $2.a = 3$. Associate the state aa of \mathcal{A}_w with the pair $(3, q_2)$, where $q_2 = 1.a = 1$. To define $2.b$ we see that the condition $\{1, 2\}.b = \{0, 1\}$ must be satisfied. Since $1.b = 0$, we put $2.b = 1$.

Step 3. Add state 4 to Q and put $3.b = 4$. Associate the state aab of \mathcal{A}_w with the pair $(4, q_3)$, where $q_3 = 1.b = 0$. To define $3.a$ we see that the condition $\{1, 3\}.a = \{1, 3\}$ must be satisfied. Since $1.a = 1$, we put $3.a = 3$.

Step 4. Add state 5 to Q and put $4.a = 5$. Associate the state $aaba$ of \mathcal{A}_w with the pair $(5, q_4)$, where $q_4 = 0.a = 2$. To define $4.b$ we see that the condition $\{0, 4\}.b = \{0, 1\}$ must be satisfied. Since $0.b = 1$, we put $4.b = 0$.

Step 5. This is the last step of the algorithm. We do not add any new states, only define the transition function on the state 5. We put $5.b = 2.b = 1$ in order to have the pair $(5, 2)$ associated with the sink state of the pair automaton. Since the condition $\{2, 5\}.a = \{1, 3\}$ must be satisfied, and $2.a = 3$, we put $5.a = 1$. The resulting pairs associated with the states of \mathcal{A}_w are shown on Fig.8. The corresponding strongly connected DFA \mathcal{B} is shown on Fig. 9.

3.2 Correctness

Now we prove the correctness of the algorithm. The proof consists of two stages. First we verify that there will be no conflict while defining the action of letters on the states of \mathcal{B} . Next we show that the resulting automaton \mathcal{B} is strongly connected.

Stage 1. Consider the word a^n . Construct the DFA \mathcal{B} using the algorithm. Its transition function is defined as follows:

$$\delta_{\mathcal{B}}(i, a) = \begin{cases} i+1 & \text{if } 1 < i < n; \\ 1 & \text{if } i = 1 \text{ or } i = n; \\ 2 & \text{if } i = 0. \end{cases} \quad \delta_{\mathcal{B}}(i, b) = \begin{cases} 1 & \text{if } i = 0 \text{ or } 1 < i \leq n; \\ 0 & \text{if } i = 1. \end{cases}$$

Obviously, the automaton is strongly connected and all assignments are correct. Thus we assume that the word w contains b , i.e. $w = a^k b v$, $v \in \Sigma^*$. We need to show that on i^{th} step of the algorithm either $q_{i-1} \cdot \overline{w[i]} = p_j$ or $q_{i-1} \cdot \overline{w[i]} = q_j$, where (p_j, q_j) is the pair associated with the prefix $w[1..j]$ of w . Since $w = a^k b v$, the states $w[1], w[1..2], \dots, w[1..k-1]$ in \mathcal{A}_w are mapped under the action of b to ε . By the construction of \mathcal{B} the state ε is associated with the pair $(0, 1)$. Thus, the letter b maps the states p_0, p_1, \dots, p_{k-1} to 0 or 1. Since the state 1 is fixed by the letter a , we have that $q_0 = 1, q_1 = 1, \dots, q_k = 1$. Since the letter b maps the state 1 to 0, we have $q_{k+1} = 0$. Note that $p_{\ell-1}$ is equal to the length of the prefix $w[1..\ell]$ of w . If v and w have common non-empty prefix, then $q_{k+2} = 2, q_{k+3} = 3$, etc. So the sequence of q 's consists of blocks of 1's and possibly blocks 0, 2, 3... corresponding to the length of some prefix. Let u be the maximal suffix of the word $w[1..i-1]$ which appears in this word as a prefix. Then in the pair (p_{i-1}, q_{i-1}) the state q_{i-1} is either $p_{|u|}$ or $q_{|u|}$. It is clear that in \mathcal{A}_w the state $|u|$ under the action of $\overline{w[i]}$ maps to j . Thus we can define the action of $\overline{w[i]}$ on the state p_{i-1} to make the pair (p_{i-1}, q_{i-1}) map to (p_j, q_j) . Indeed, if $q_{i-1} = p_{|u|}$ then $p_{i-1} \cdot \overline{w[i]} = q_{|u|} \cdot \overline{w[i]}$, otherwise put $p_{i-1} \cdot \overline{w[i]} = p_{|u|} \cdot \overline{w[i]}$.

Stage 2. Now we prove that the resulting automaton \mathcal{B} is strongly connected. Denote by \mathcal{A}'_w the minimal automaton \mathcal{A}_w after deleting the sink state. Let us assume first that \mathcal{A}'_w is strongly connected. Let i be an arbitrary state in the automaton \mathcal{B} . By assumption there is a path in $\mathcal{P}^{[2]}(\mathcal{B})$ from the pair (i, q_{i-1}) (associated with some state in \mathcal{A}_w) to the pair $(0, 1)$ (associated with the state ε). Hence, either 0 or 1 is reachable from the state i . In fact both states 0 and 1 are reachable from i , since $0.b = 1$ and $1.b = 0$. By the construction the pair (i, q_{i-1}) is reachable from the pair $(0, 1)$, so the state i is reachable both from 0 and 1 in \mathcal{B} . Thus, in this case the automaton \mathcal{B} is strongly connected.

Let us prove that if $w \notin \{a^{n-1}b, ab^{n-1}\}$, then the automaton \mathcal{A}'_w is strongly connected. Consider an arbitrary state $w[1..i]$ of \mathcal{A}'_w . This state is obviously reachable from the state ε . We show that the state ε is reachable from $w[1..i]$. Let $w[1..j] = w[1..i].c^{|w|}$, where $c = \overline{w[i+1]}$. The state $w[1..j]$ is a maximal prefix of w of the form c^k . If $c = b$, then $w[1..j] = \varepsilon$, so we are done. Suppose $c = a$. Apply b to the state $w[1..j]$ (this is possible, since $w \neq a^{n-1}b$). Next we apply $\overline{w[j+2]}$. If $\overline{w[j+2]} = b$, then $w[1..j].bb = \varepsilon$. If $\overline{w[j+2]} = a$, then $w[1..j].bab = \varepsilon$ (since $w \neq ab^{n-1}$). So in both cases the state ε is reachable from $w[1..i]$.

It remains to apply the algorithm to $w = a^{n-1}b$ and $w = ab^{n-1}$ to make sure that the resulting automata in this case are also strongly connected. Indeed, in case $w = a^{n-1}b$ the transitions of the automaton \mathcal{B} are defined as follows:

$$\delta_{\mathcal{B}}(i, a) = \begin{cases} i+1 & \text{if } 1 < i < n, \\ i & \text{if } i = 1 \text{ if } i = n, \\ 2 & \text{if } i = 0, \end{cases} \quad \delta_{\mathcal{B}}(i, b) = \begin{cases} 1 & \text{if } i = 0 \text{ if } 1 < i < n, \\ 0 & \text{if } i = 1 \text{ if } i = n. \end{cases}$$

The states $0, 2, \dots, n$ form a cycle marked by the word $a^{n-1}b$, furthermore $0.b = 1$ and $1.b = 0$. Thus, \mathcal{B} is strongly connected.

In case $w = ab^{n-1}$ the algorithm constructs the following DFA \mathcal{B} :

$$\delta_{\mathcal{B}}(i, a) = \begin{cases} 1 & \text{if } 0 \leq i \leq n \text{ is odd,} \\ 2 & \text{if } 0 \leq i \leq n \text{ is even,} \end{cases} \quad \delta_{\mathcal{B}}(i, b) = \begin{cases} i+1 & \text{if } 0 \leq i < n, i \neq 1, \\ 0 & \text{if } i = 1, \end{cases}$$

$$\delta_{\mathcal{B}}(n, b) = \begin{cases} 0 & \text{if } n \text{ is even,} \\ 1 & \text{if } n \text{ is odd.} \end{cases}$$

If n is even, then the states $0, 2, \dots, n$ form a cycle marked by the word ab^{n-1} , if n is odd, then the states $0, 2, \dots, n, 1$ form a cycle marked by the word ab^n . Thus, \mathcal{B} is strongly connected.

4 On syntactic semigroup of a principal ideal language

In the previous section for each word w of length n we constructed a strongly connected synchronizing DFA \mathcal{B} with $n+1$ states such that $\text{Syn}(\mathcal{B}) = \Sigma^*w\Sigma^*$. Is it possible to construct such a DFA with less than $n+1$ states? In [4] it

was shown that for the case $w \in \{a^n, b^n, a^{n-1}b\}$ we have $rc(L_w) = n + 1$, where $L_w = \Sigma^* w \Sigma^*$ and $|w| = n$. But in general this question remains open. Nevertheless computer experiments show that the answer seems to be negative, i.e. the minimal in terms of reset complexity strongly connected synchronizing DFA has $n + 1$ states. Another observation concerns the structure of that DFA. Even for a word w of length 3 there may be several non-isomorphic strongly connected synchronizing automata yielding the minimum of reset complexity. But, as experiments show, the *transition semigroups* of all these automata have the same algebraic structure. In this regard it is interesting to study the structure of the transition semigroup of a synchronizing automaton for which given ideal language serves as the language of synchronizing words.

For an ideal language $L \subseteq \Sigma^*$ the *Myhill congruence* [5] \approx_L of L is defined as follows:

$$u \approx_L v \text{ if and only if } xuy \in L \Leftrightarrow xvy \in L \text{ for all } x, y \in \Sigma^*.$$

This congruence is also known as *the syntactic congruence* of L . The quotient semigroup Σ^+ / \approx_L of the relation \approx_L is called the *syntactic semigroup* of L .

Proposition 1. *Let L be an ideal language, S the syntactic semigroup of L and $S(\mathcal{B})$ the transition semigroup of a synchronizing DFA \mathcal{B} for which L serves as the language of synchronizing words. Then S is a homomorphic image of $S(\mathcal{B})$.*

Proof. Take an arbitrary word $x \in \Sigma^*$. Let $[x]$ be the class of x in S , and $\{x\}$ the class of x in $S(\mathcal{B})$. Define the map $f : S(\mathcal{B}) \rightarrow S$ by the rule $f(\{x\}) = [x]$. Check that f is a homomorphism. First, we check the correctness of defining f . Consider two words u and v from the same class in $S(\mathcal{B})$, it means that $\{u\} = \{v\}$. Show that in this case $[u] = [v]$. We need to check that for any $x, y \in \Sigma^*$ from $xuy \in L$ it follows that $xvy \in L$. By conditions of the proposition $L = \text{Syn}(\mathcal{B})$. Since $\{u\} = \{v\}$, then u and v generate the same transformations in the transition semigroup of \mathcal{B} . But then either both xuy and xvy synchronize \mathcal{B} , or both do not synchronize. Hence $[u] = [v]$. And finally, $f(\{u\}\{v\}) = f(\{uv\}) = [uv] = [u][v] = f(\{u\})f(\{v\})$. This completes the proof. \square

Note that this proposition holds for every regular ideal language L . It means that if the transition semigroup of any DFA \mathcal{B} such that $\text{Syn}(\mathcal{B}) = L$ possesses some algebraic property which is preserved under homomorphisms, then also the syntactic semigroup of L must possess this property. Thereby it is interesting to study the structure of the syntactic semigroup of an ideal language. The *syntactic complexity* $\sigma(L)$ of a regular language L is the cardinality of its syntactic semigroup. The notion of syntactic complexity is studied quite extensively: for surveys of this topic and lists of references we refer the reader to [1, 3]. In [1] it was conjectured that in case of ideal languages $\sigma(L) \leq n^{n-2} + (n-2)2^{n-2} + 1$, where n is the state complexity of L . Also in [1] it was shown that there exists an ideal language of syntactic complexity $n^{n-2} + (n-2)2^{n-2} + 1$. We consider partial case of principal ideal languages. Recall that $u \in \Sigma^+$ is an *inner factor* of w if there exist words $t, s \in \Sigma^+$ such that $w = tus$. Denote by $N(w)$ the number of different inner factors of w . We prove the following

Theorem 1. *Let $w \notin \{a^{n-1}b, ab^{n-1}, ba^{n-1}, b^{n-1}a\}$ and $L = \Sigma^*w\Sigma^*$, where $|w| = n$. Then $\sigma(L) = n^2 + 1 + N(w)$.*

Proof. Build the minimal automaton \mathcal{A}_L recognizing L . We refer to words $u \in \Sigma^*$ as pairs (s, p) , where s is the maximal suffix of w that appears in u as a prefix, and p is the maximal prefix of w that is also a suffix of u . For instance, consider the word $w = aabab$. For the word $u = abbaba$ the corresponding pair is (ab, a) . Assume that $s \neq w$ and $p \neq w$.

First we show that the words corresponding to different pairs (s_1, p_1) and (s_2, p_2) define different transformations of the DFA \mathcal{A}_L . Indeed, if $p_1 \neq p_2$, then the first (initial) state of \mathcal{A} is mapped by this prefixes into different states (p_1 and p_2 respectively). Thus, the corresponding transformations act differently on the first state. If $s_1 \neq s_2$, then without loss of generality assume that $|s_1| \leq |s_2|$. In this case there exists a prefix-state, which s_2 maps to the terminal state, and s_1 does not. Thus, again the corresponding transformations act differently on this state. If in pairs (s_1, p_1) and (s_2, p_2) we have that $s_1 = s_2 = w$ or $p_1 = p_2 = w$, then all words corresponding to such pairs generate the same transformation, since $w \in \text{Syn}(\mathcal{A})$.

We construct for each pair (s, p) (s or p also can be empty words) a word in the syntactic semigroup which differs from w . Take the suffix s , append to it $2 \cdot |w| - |s|$ times the letter, different from the last letter of s . Denote this letter by \bar{x} . If $s = \varepsilon$, then \bar{x} is chosen to be different from the last letter of w . Further we append $2 \cdot |w| - |p|$ times the letter, different from the first letter of w . Denote this letter by \bar{y} , then complete the word by adding p to the end. By the construction s is the maximal suffix of w that appears in u as a prefix, and p is the maximal prefix of w , which is also a suffix of u . However, it could happen that after adding blocks of \bar{x} and \bar{y} the word w appeared in u . It would be only in the case when $w = \bar{x}^l \bar{y}^m$ for some $l, m \geq 1$. Assume that $l, m > 1$, because $w \notin \{a^{n-1}b, ab^{n-1}, ba^{n-1}, b^{n-1}a\}$. For such a word w we construct u as follows. First append the letter \bar{x} as above. Then add the word $\bar{y}\bar{x}$, and then carry on the construction as described above. The word w is not a factor of the constructed word u . So if $w \notin \{a^{n-1}b, ab^{n-1}, ba^{n-1}, b^{n-1}a\}$, then the syntactic semigroup of L consists of at least $n^2 + 1$ elements.

If words u referred to the pair (s, p) have length at least $4n$, then these words define the same transformation of \mathcal{A}_L . Otherwise, applying the word u to some state q we may obtain another prefix t :

$$\underbrace{\overbrace{qu \dots}^w}_t$$

This situation occurs only when u is an inner factor of w . So we need to add in the transition semigroup all inner factors of w .

And finally the whole word w always belongs to the transition semigroup. \square

Note that for the word $a^{n-1}b$ it is impossible to construct words corresponding to pairs of the form (ε, p) or (b, p) , where p is a prefix w . In this case $\bar{x} = a$,

$\bar{y} = b$. And the word u constructed by the algorithm from the Theorem contains w as a factor. Hence we need to find the words corresponding to the pairs of this form separately. There is no word in the transition semigroup corresponding to the pair $(\varepsilon, \varepsilon)$. Pairs (ε, p) can be associated with the corresponding prefixes p , and pairs (b, p) with the word bp . All factors of the word has already been considered, because they coincide with prefixes and suffixes. Finally, $\sigma(L) = 1 + n^2 - 2n + 2n - 1 = n^2$.

Analogously, the same result holds for words ab^{n-1} , $b^{n-1}a$, and ba^{n-1} . The number of all different inner factors is estimated as $N(w) \leq \frac{(n-1)(n-2)}{2}$. And we have the following estimation: $n^2 \leq \sigma(L) \leq 1.5n^2 + o(n^2)$. In particular, lower bound is tight. Moreover, the equality $\sigma(L) = n^2$ holds only for words $a^{n-1}b$, ab^{n-1} and $b^{n-1}a$, ba^{n-1} .

The upper bound of the value $\sigma(L)$ is $1.5n^2 + o(n^2)$. Next we give an example of the language $L_w = \Sigma^* w \Sigma^*$ for which the equality $\sigma(L_w) = 1.5n^2 + o(n^2)$ takes place.

Proposition 2. *There exists a word w of length $|w| = n \geq 21$ for which $\sigma(L_w) = 1.5n^2 + o(n^2)$.*

Proof. We prove the Proposition in a constructive way. Take the word $w = ab^2a^3b^4 \dots a^{k-1}b^k$, i.e. $n = 1 + 2 + \dots + k$ for even $k \geq 4$. Count $N(w)$, the number of all different inner factors v of the word w . Denote by m the maximal power of b that appears in v , i.e. $v = tb^m r$.

If $m = 0$, then v does not contain b , there are $k - 1$ such factors.

If $m = 1$, then we have three cases:

$v = ba^l$ ($0 \leq l \leq k - 1$);

$v = a^l b$ ($1 \leq l \leq k - 1$);

$v = ba^l b$ (l is odd from 3 to $k - 1$).

There are $k + k - 1 + \frac{k-2}{2} = \frac{5}{2}k - 2$ such factors.

If $m = 2$, then we have three cases:

$v = b^2 a^l$ ($0 \leq l \leq k - 1$);

$v = a^l b^2$ ($1 \leq l \leq k - 1$);

$v = b^2 a^l b$, or $v = ba^l b^2$, or $v = b^2 a^l b^2$ (l is odd from 3 to $k - 1$).

There are $k + k - 1 + 3\frac{k-2}{2} = \frac{7}{2}k - 4$ such factors.

If $m = k - 1$, then $v = tb^{k-1}$, where $0 \leq |t| \leq n - k - 1$. There are $n - k$ such factors.

Consider the case of odd m . Let $2 < m < k - 1$. We have the following cases:

$v = tb^m$, $0 \leq |t| \leq 2 + 3 + \dots + m = \frac{(2+m)(m-1)}{2}$;

$v = b^m a^l$, $1 \leq l \leq k - 1$;

$v = b^m a^l b^r$, $1 \leq r \leq m$, $m + 2 \leq l \leq k - 1$, l is odd;

$v = a^l b^m$, $m + 1 \leq l \leq k - 1$;

$v = b^r a^l b^m$, $1 \leq r \leq m - 1$, $m + 2 \leq l \leq k - 1$, l is odd.

There are $\frac{m(m+1)}{2} + k - 1 + \frac{k-m-1}{2}m + k - 1 - m + \frac{k-m-1}{2}(m - 1)$ such factors.

Consider the case of even m . Let $2 < m < k - 1$. The following cases are possible:

$v = tb^m, 0 \leq |t| \leq 2 + 3 + \dots + m - 1 = \frac{(1+m)(m-2)}{2};$
 $v = b^m a^l, 1 \leq l \leq k - 1;$
 $v = b^m a^l b^r, 1 \leq r \leq m, m + 1 \leq l \leq k - 1, l \text{ is odd};$
 $v = a^l b^m, m \leq l \leq k - 1;$
 $v = b^r a^l b^m, 1 \leq r \leq m - 1, m + 1 \leq l \leq k - 1, l \text{ is odd};$
 $v = tb^m a^l, 1 \leq |t| \leq 2 + 3 + \dots + m - 1 = \frac{(1+m)(m-2)}{2} \text{ and } 1 \leq l \leq m + 1;$
 $v = tb^m a^{m+1} b^r, 1 \leq |t| \leq 2 + 3 + \dots + m - 1 = \frac{(1+m)(m-2)}{2} \text{ and } 1 \leq r \leq m.$
 There are $\frac{m(m-1)}{2} + k - 1 + \frac{k-m}{2}m + k - m + \frac{k-m}{2}(m-1) + \frac{(m+1)^2(m-2)}{2}(m -$
 1) + $\frac{m(m+1)(m-2)}{2}$ such factors.

Finally, the total number of all different inner factors is equal to

$$\begin{aligned}
 N(w) = & k - 1 + \frac{5}{2}k - 2 + \frac{7}{2}k - 4 + n - k + \sum_{m=3, m \text{ is odd}}^{k-3} \left(\frac{m(m+1)}{2} + \right. \\
 & \left. + k - 1 + \frac{k-m-1}{2}m + k - 1 - m + \frac{k-m-1}{2}(m-1) \right) + \\
 & + \sum_{m=4, m \text{ is even}}^{k-2} \left(\frac{m(m-1)}{2} + k - 1 + \frac{k-m}{2}m + k - m + \frac{k-m}{2}(m-1) + \right. \\
 & \left. + \frac{(m+1)^2(m-2)}{2}(m-1) + \frac{m(m+1)(m-2)}{2} \right) = n + 6k - 7 + \\
 & + \underbrace{\sum_{m=3}^{k-3} \left(\frac{m(m-1)}{2} + \frac{k-m}{2}(2m-1) + 2k - m - 1 \right)}_{N_1(k)} + \\
 & + \underbrace{\sum_{m=3, m \text{ odd}}^{k-3} \left(-\frac{1}{2} \right)}_{N_2(k)} + \underbrace{\sum_{m=4, m \text{ even}}^{k-2} \left(m^3 - \frac{m^2}{2} - \frac{5m}{2} - 1 \right)}_{N_3(k)}.
 \end{aligned}$$

Count the values $N_1(k)$, $N_2(k)$, $N_3(k)$. It is easy to find $N_2(k) = -\frac{1}{2} \cdot \frac{k-4}{2} = -\frac{k-4}{4}$. Then using formula $2^3 + 4^3 + \dots + (2n)^3 = 2n^2(n+1)^2$ and $2^2 + 4^2 + \dots + (2n)^2$ obtain $N_3(k) = \frac{k^4}{8} - \frac{7}{12}k^3 + \frac{k^2}{8} + \frac{7}{12}k + 1$. Finally, using the formula $1^2 + 2^2 + \dots + n^2 = \frac{1}{3}n^2 + \frac{1}{2}n^2 + \frac{1}{6}n$, find $N_1(k) = \frac{k^3}{3} + \frac{k^2}{4} - \frac{103}{12}k + 9$. Generalizing all results obtain $\sigma(L) = n^2 + 1 + n + 6k - 7 + N_1(k) + N_2(k) + N_3(k) = n^2 + n + \frac{k^4}{8} - \frac{k^3}{4} + \frac{3}{8}k^2 - \frac{9}{4}k + 5$. Note that $n = 1 + 2 + \dots + k = \frac{k^2+k}{2}$. Then $\sigma(L) = \frac{3}{2}n^2 + \frac{5}{2}n - kn - 3k + 5$, where k is a positive root of the equation $k^2 + k - 2n = 0$.

Let $n \geq 21$ can not be decomposed into a sum $n = 1 + 2 + \dots + k$ for some even k . Find the number k , such that $1 + 2 + \dots + k < n \leq 1 + 2 + \dots + k + 1$. Consider the word $w = ab^2a^3 \dots b^k a^l$, where $l = n - \frac{1+k}{2}k$. Otherwise, $1 + 2 + \dots + k + 1 < n < 1 + 2 + \dots + k + 2$, then construct the word $w = ab^2a^3 \dots b^k a^{k+1} b^r$, where $r = n - \frac{2+k}{2}(k+1)$. Using analogous arguments one may verify that the equality $\sigma(L) = 1.5n^2 + o(n^2)$ holds. \square

The previous theorem provides rather simple formula to calculate the syntactic complexity of a principal ideal language. Indeed, we do not need to construct the minimal automaton of the language and then analyze its transition semigroup for an arbitrary w . We just need to count all different inner factors of w and it can be done in time $O(n^2)$ in a trivial way.

Acknowledgement. The authors acknowledge support from the Presidential Programm for young researchers, grant MK-266.2012.1.

References

1. Brzozowski J., Ye Y. *Syntactic Complexity of Ideal and Closed Languages*. In G. Mauri, A. Leporati (Eds.) Proc. DLT 2011, Lect. Notes Comp. Sci. Vol. 6795, Springer-Verlag Berlin-Heidelberg 2011. P. 117–128.
2. J. Černý. *Poznámka k homogénnym experimentom s konečnými automatami.*, Mat.-Fyz. Cas. Slovensk. Akad. Vied. **14** (1964) 208–216.
3. Holzer, M., König, B.: *On deterministic finite automata and syntactic monoid size*. Theoret. Comput. Sci. **327** (2004) P. 319–347
4. Maslennikova M.I. *Reset Complexity of Ideal Languages*. In M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, R. Špánek, G. Turán (eds.) Int. Conf. SOFSEM 2012, Proc. Volume II, Institute of Computer Science Academy of Sciences of the Czech Republic, 2012, P. 33–44.
5. Myhill, J.: *Finite automata and representation of events*. Wright Air Development Center Technical Report, 57624 (1957)
6. D. Perrin *Finite automata*. Handbook of Theoretical computer Science, J. van Leeuwen, (ed.), Elsevier, B., P. 1–57, 1990.
7. E. Pribavkina, E. Rodaro. *Synchronizing automata with finitely many minimal synchronizing words*// Inf. and Comput. V.**209**(3), 2011. P.568–579.
8. E. Pribavkina, E. Rodaro. *Recognizing synchronizing automata with finitely many minimal synchronizing words is PSPACE-complete*// B. Löwe, D. Normann, I. Soskov, A. Soskova (Eds.) Proc. CiE 2011, Lect. Notes Comp. Sci. Vol. 6735, Springer-Verlag Berlin-Heidelberg 2011. P. 230–238.
9. M. V. Volkov. *Synchronizing automata and the Černý conjecture*, in C. Martín-Vide, F. Otto, H. Fernau (eds.), Languages and Automata: Theory and Applications. LATA 2008. Lect. Notes Comp. Sci. **5196**, Berlin, Springer (2008) 11–27.
10. M. V. Volkov *Synchronizing automata preserving a chain of partial orders*// In J. Holub and J. Ždárek (eds.) Implementation and Application of Automata. Proc. 12th Int. Conf. CIAA 2007, Lect. Notes Comp. Sci., Springer-Verlag, Berlin-Heidelberg-New York. 2007. V.4783. P.27–37.